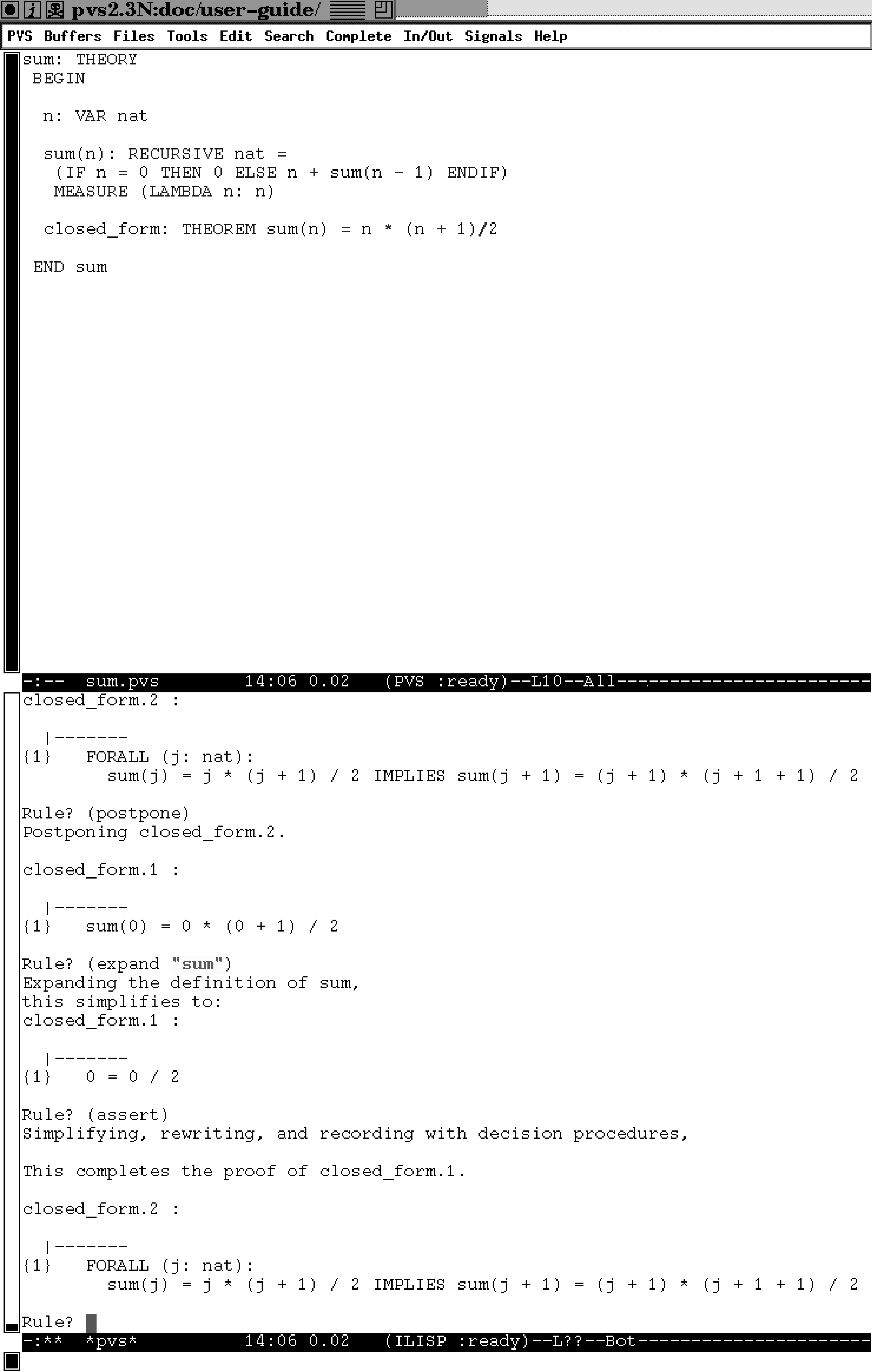
pvs2.3N:doc/user-guide/
PVS Buffers Files Tools Edit Search Complete In/Out Signals Help

```
sum: THEORY
 BEGIN

  n: VAR nat

  sum(n): RECURSIVE nat =
   (IF n = 0 THEN 0 ELSE n + sum(n - 1) ENDIF)
   MEASURE (LAMBDA n: n)

  closed_form: THEOREM sum(n) = n * (n + 1)/2

 END sum
```

```
-:--  sum.pvs         14:06 0.02    (PVS :ready)--L10--All----------------------
```

```
closed_form.2 :

  |-------
{1}   FORALL (j: nat):
        sum(j) = j * (j + 1) / 2 IMPLIES sum(j + 1) = (j + 1) * (j + 1 + 1) / 2

Rule? (postpone)
Postponing closed_form.2.

closed_form.1 :

  |-------
{1}   sum(0) = 0 * (0 + 1) / 2

Rule? (expand "sum")
Expanding the definition of sum,
this simplifies to:
closed_form.1 :

  |-------
{1}   0 = 0 / 2

Rule? (assert)
Simplifying, rewriting, and recording with decision procedures,

This completes the proof of closed_form.1.

closed_form.2 :

  |-------
{1}   FORALL (j: nat):
        sum(j) = j * (j + 1) / 2 IMPLIES sum(j + 1) = (j + 1) * (j + 1 + 1) / 2

Rule?
```

```
-:**  *pvs*           14:06 0.02    (ILISP :ready)--L??--Bot----------------------
```